Another approximation algorithm – Weighted Set Cover

Given: A universe $\mathcal{U}$ of n elements, and a collection $S_1, \ldots, S_m$ of m subsets of $\mathcal{U}$, and a non-negative weight w_i associated with each S_i.

Find: A collection C of subsets that cover all of $\mathcal{U}$ with the smallest total weight:

$$W_C = \sum_{S_i \in C} w_i$$

Note: This is at least as hard as the decision version of set cover (here, sets can have arbitrary weight $\neq 1$).

Algorithmic Intuition: Good sets have

(1) Small weight (w_i)

(2) Cover many elements ($|S_i|$ is large)

Pursuing either one in isolation can lead to very poor solutions.

Instead, we'll search for sets with a small value of

$$b_i(R) = w_i / |S_i \cap R|$$

where R is the remaining subset of $\mathcal{U}$ (i.e. the subset of $\mathcal{U}$ that remains uncovered).

This selects sets that will yield the <u>maximum marginal benefit</u>.

So, our algorithm will look like:

Greedy WSC:

$C = \emptyset$

while $|U - C| > 0$:

$R = U - C$

Select S_i with the largest $b_i(R)$

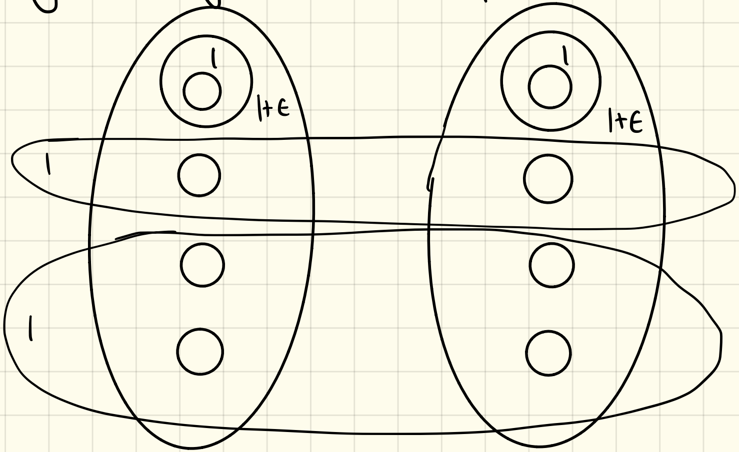
$C = C \cup S_i$

End

Return C

To make this practically efficient, we will want to maintain the sets in some sort of priority queue, keyed on their benefit. This is a bit tricky, because after each new set is selected, the benefit of many other sets might change. However, there exist good data structures (e.g. Fib heaps) that yield a practically efficient implementation of GreedyWSC.

Clearly Greedy WSC isn't optimal. Can we construct a bad instance?



Greedy WSC here will choose a cover of weight 4, while $2+2\epsilon$ is optimal. Let's stop and think for a moment why this is the case.

Finding a meaningful lower bound here is harder than in the load-balancing problem

Let $c_s = w_i / |S_i \cap R|$ for all $s \in S_i \cap R$ be the cost paid "per-element" when it's covering set was chosen.

We have: If C is the set cover obtained by GreedyWSC, then $\sum_{S_i \in C} w_i = \sum_{s \in U} c_s$ ... that is, the cost of the cover is simply the sum of the costs that we spent on each element in U when it was covered.

The key will be to upper-bound the ratio:

$$\left(\sum_{s \in S_k} c_s \right) \Big/ w_k$$

An optimal solution must cover the full universe via the sets it contains, so we seek a bound on the weight it must use.

We will show:

Lemma: For every set S_k, the sum $\sum_{s \in S_k} c_s$ is <u>at most</u> $H(|S_k|) \cdot w_k$.

Where $H(n) = \sum_{i=1}^{n} \frac{1}{i}$ is the harmonic function.

Proof: Assume the elements of S_k are the first $d = |S_k|$ elements of U so $S_k = \{s_1, s_2, \ldots, s_d\}$. Assume they are labeled in the order in which they are assigned a cost c_{s_j} by the algorithm.

Consider the iteration in which s_j is covered for some $j \leq d$. At this iteration, $s_j, s_{j+1}, \ldots, s_d \in R$. So $|S_k \cap R| \geq d-j+1$, and the average cost of S_k is at most

$$\frac{w_k}{|S_k \cap R|} \leq \frac{w_k}{d-j+1}$$

Not necessarily equality since s_j may be covered in the same iteration as some j' with $j' < j$.

In this iteration, greedy WSC selected S_i with minimum average cost, so the average cost of S_i is at most that of S_k. It is the average cost of S_i that is assigned to s_j, so:

$$c_{s_j} = \frac{w_i}{|S_i \cap R|} \leq \frac{w_k}{|S_k \cap R|} \leq \frac{w_k}{d-j+1}$$

Adding such a bound for all elements, we get

$$\sum_{s \in S_k} c_s = \sum_{j=1}^{d} c_{s_j} \leq \sum_{j=1}^{d} \frac{w_k}{d-j+1} = \frac{w_k}{d} + \frac{w_k}{d-1} + \cdots + \frac{w_k}{1}$$

$$= H(d) \cdot w_k$$

This leads to

Theorem (Greedy Weighted Set Cover): The set cover C selected by Greedy WSC has weight at most $H(d^*)$ times the optimal weight w^*, where $d^* = \max_i |S_i|$

Proof: Let C^* be an optimal cover so that $C^* = \sum_{S_i \in C} w_i$. For each S_i:

$$w_i \geq \frac{1}{H(d^*)} \sum_{s \in S_i} c_s$$

because this is a valid set cover (all of $\mathcal{U}$ is covered) we have

$$\sum_{S_i \in C^*} \sum_{s \in S_i} c_s \geq \sum_{s \in \mathcal{U}} c_s$$

with our lemma, this gives:

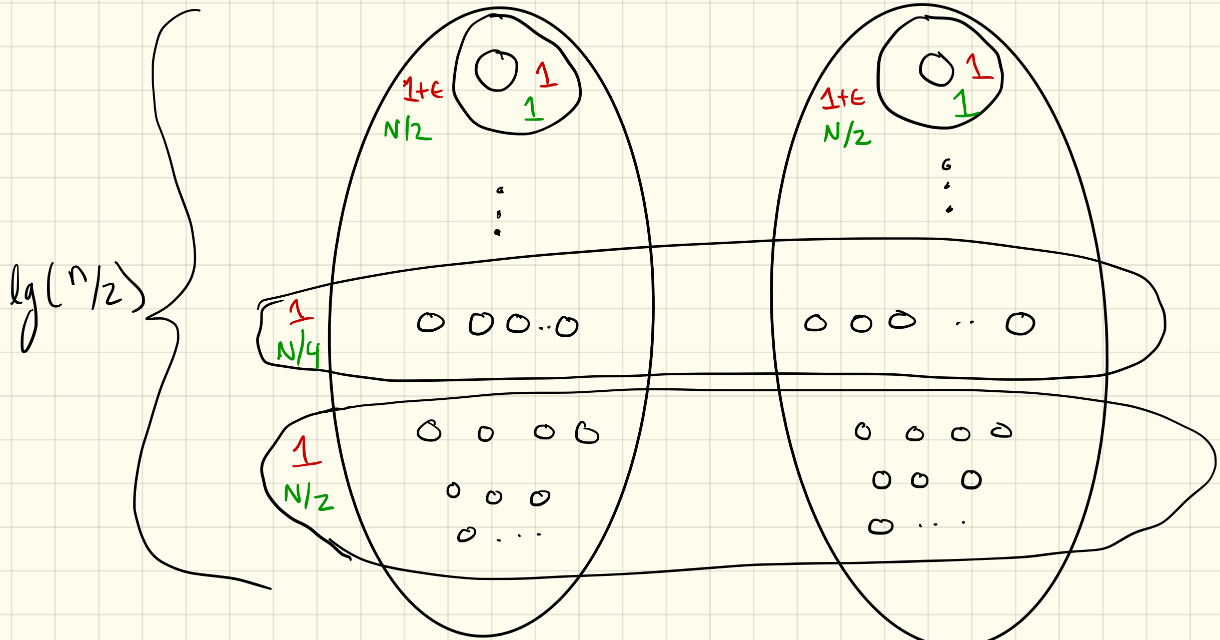
$$w^* = \sum_{S_i \in C^*} w_i \geq \sum_{S_i \in C^*} \frac{1}{H(d^*)} \sum_{s \in S_i} c_s \geq \frac{1}{H(d^*)} \sum_{s \in \mathcal{U}} c_s = \frac{1}{H(d^*)} \sum_{S_i \in C} w_i$$

Note: $H(d^*) = \Theta(\log d^*)$ so the factor here is up to logarithmic in d^*!

However, d^* is the size of the largest set. If this size is small, this can be much better than $\log(n) \cdot OPT$.

Interestingly, it has been shown that no polynomial time approximation scheme (PTAS) can achieve a lower bound much better than $c \cdot l(n)$ for some fixed constant c, unless $P = NP$.

How does a bad example for Greedy WSC look?



Approximation algorithms under reductions:

General properties

1) Approximation ratios do not generally reduce i.e. if one has a c-factor approx for problem A and reduces B to A, this does not imply c-factor approx for B.

2) Sometimes approx ratios do reduce, but you have to <u>prove this</u>

3) If a c-factor approx is optimal for A, and it <u>does</u> reduce to B, that does not imply it is optimal for B.

Very Basic types of approximations

non-constant factor (as with Set Cover... approx badness grows with problem)

constant factor (can get within $c \cdot OPT$)

arbitrary (can get within $(1+\epsilon) \cdot OPT$) for any ϵ.
$\hookrightarrow$ Polynomial time approximation scheme PTAS.

Let's look at a case where the approx does reduce.

Set Cover $\rightarrow$ Vertex Cover

Weighted Vertex Cover - Given a graph $G=(V,E)$ and a weight w_i for $i \in V$, find a vertex cover of G with the smallest

$$W = \sum_{i \in C} w_i$$

Does set cover approx carry over to VC?

Theorem: We can use the approximation algo from weighted set cover to give an $H(d)$-approximation for weighted vertex cover where d is the maximum degree of a vertex in G.

Proof: Consider an instance of weighted Vertex Cover specified by $G = (V, E)$. Define an instance of weighted Set cover as in our previous reduction.

- Let $U = E$
- For each $v_i \in V$ define S_i where $S_i = \{ \underbrace{\{u, v_i\}}_{\text{edges adjacent to } v_i} \mid \{u, v_i\} \in E \}$
- Let $\text{weight}(S_i) = w_i$ = weight of vertex v_i

Note, the maximum size of any S_i is exactly the maximum degree of any vertex.

It follows immediately that a weighted set cover of weight

$W = \sum_{S_i \in C} w_i$ yields a corresponding weighted vertex cover of equivalent weight.

Can we do better? Yes.

But first, a cautionary example of where an approx isn't preserved.

Consider Vertex Cover and Independent Set. Recall

- Independent Set $\leq_p$ Vertex Cover

Does our approx for VC imply an approx for IS? NO!

Recall $I \subseteq V$ is an independent set iff $S = V - I$ is a vertex cover. Given a minimum size vertex cover S^*, we do in fact obtain a maximum size Independent set as $I^* = V - S$.

Suppose we use an approx alg for VC to get an approximately minimum VC S. Then $I = V - S$ is an independent set, but it does not preserve the approximation factor.

Suppose e.g. that S^* and I^* are both of size $|V|/2$. If we have a 2 approx for VC, we could obtain an answer $S = V$, but then the corresponding approx independent set $I = V - S = V - V = \emptyset$ has no elements ... the factor 2 approximation isn't preserved.

A better approximation algo for weighted vertex cover.

We'll consider what is called a "pricing" method (ch 11.4)

Intuition:

- w_i is the "cost" for using i in the vertex cover
- Each edge is an "agent" who is willing to "pay" something to the node that covers it.
- Algo: in addition to finding a cover, we will determine "prices" $p_e \geq 0$, so that if each edge e pays p_e, it will approximately cover the cost of set S.

- We want these prices to have a "fairness" property:

$$\sum_{e=(i,j)} p_e \leq w_i$$

These fair prices will provide a lower bound on the cost of any solution.

Theorem: For any vertex cover S^*, and any non-negative fairness prices p_e, we have $\sum_{e \in E} p_e \leq w(S^*)$.

Proof: Consider a VC S^*. By definition of fairness, we have $\sum_{e=(i,j)} p_e \leq w_i$ for all nodes $i \in S^*$. Adding these inequalities over all nodes in S^*, we get:

$$\sum_{i \in S^*} \sum_{e=(i,j)} p_e \leq \sum_{i \in S^*} w_i = w(S^*)$$

The LHS is a sum of edge prices p_e. Since S^* is a VC, each edge contributes at least one p_e term to the LHS, but e could be covered from both sides. However, prices are non-negative and so the sum of the LHS is at least as large as the sum of all p_e i.e.

$$\sum_{e \in E} p_e \leq \sum_{i \in S^*} \sum_{e=(i,j)} p_e$$

combining with the previous inequality, we have

$$\sum_{e \in E} p_e \leq w(S^*)$$

Algo:

Approx VC(G, w):

Set $p_e = 0 \ \forall e \in E$

While there is an $e=(i,j)$ such that neither i nor j is "tight":

Increase p_e without violating fairness

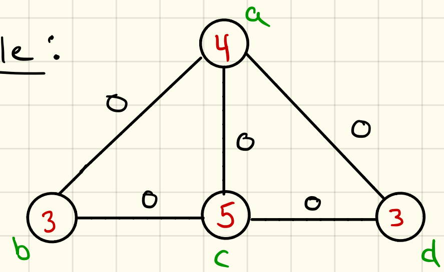
END

Let S be the set of all tight nodes

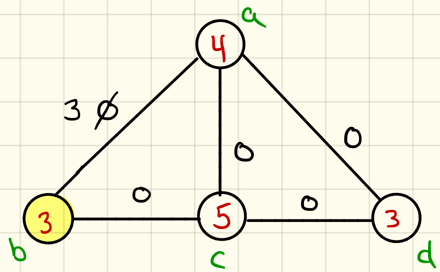
Return S

- A node is "tight" (paid for) if $\sum_{e=(i,j)} p_e = w_i$

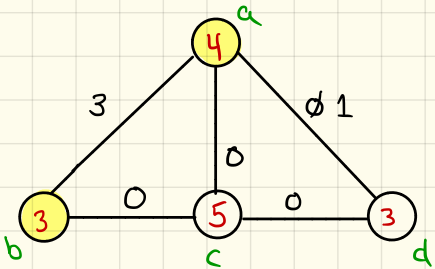
Example:



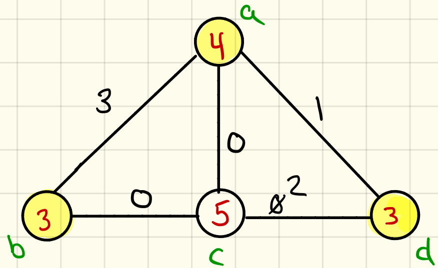
Step 1



Step 2



Step 3



Step 4

Now, every edge is adjacent to a "tight" node. Return {a, b, d} as the cover.

Note: this is not optimal as we could cover with {a, c} of weight 9.

So, why is this a lower bound and not a tight bound (i.e. why can the edges not fully pay for a VC?).

The problem is that an edge can be adjacent to more than one vertex in the vertex cover, and so a given edge can pay for more than one vertex. We'll show that we cannot overpay too much. Specifically,

Theorem: The set S and prices p returned by the algorithm satisfy

$$w(S) \le 2 \sum_{e \in E} p_e$$

Proof: All nodes in S are tight, so $\sum_{e=(i,j)} p_e = w_i$ for all $i \in S$.

Summing over S we have,

$$w(S) = \sum_{i \in S} w_i = \sum_{i \in S} \sum_{e=(i,j)} p_e$$

An edge $e=(i,j)$ can be included on the RHS at most twice (if both i + j are in S), so

$$w(S) = \sum_{i \in S} \sum_{e=(i,j)} p_e \le 2 \sum_{e \in E} p_e$$

Theorem: The Set S returned by our algo is a VC, and it's cost is at most twice the cost of <u>any</u> VC.

<u>Proof</u>: First, note S is a VC. Suppose, by contradiction, that S does not cover some $e=(i,j)$. This implies that neither i nor j is tight, and is a contradiction for the termination condition of our while loop.

Let p be the prices set by our algo, and let S^* be an optimal VC. We already have

$$2\sum_{e\in E} p_e \geq w(S) \qquad \text{and} \qquad \sum_{e\in E} p_e \leq w(S^*)$$

So, the sum of the edge prices is a lower bound on the weight of <u>any</u> vertex cover, and twice the sum of the edge prices is an upper-bound on the weight of the VC our algorithm finds. Hence:

$$w(S) \leq \sum_{e\in E} p_e \leq 2\, w(S^*)$$